

PREPRINT VERSION

An agent-driven semantical identifier using radial basis neural networks and reinforcement learning

C. Napoli, G. Pappalardo, and E. Tramontana

PUBLISHED ON: **Proceedings of the XV Workshop "Dagli Oggetti agli Agenti"**

BIBITEX:

```
@inproceedings{Napoli2014Anagent
year={2014},
issn={1613-0073},
url={http://ceur-ws.org/Vol-1260/},
booktitle={Proceedings of the XV Workshop "Dagli Oggetti agli Agenti"},
title={An agent-driven semantical identifier using radial basis neural networks and reinforcement learning},
publisher={CEUR-WS},
volume={1260},
author={Napoli, Christian and Pappalardo, Giuseppe and Tramontana, Emiliano},
}
```

Published version copyright © 2014

UPLOADED UNDER SELF-ARCHIVING POLICIES

An agent-driven semantical identifier using radial basis neural networks and reinforcement learning

Christian Napoli, Giuseppe Pappalardo, and Emiliano Tramontana

Department of Mathematics and Informatics

University of Catania, Viale A. Doria 6, 95125 Catania, Italy

{napoli, pappalardo, tramontana}@dmi.unict.it

Abstract—Due to the huge availability of documents in digital form, and the deception possibility raise bound to the essence of digital documents and the way they are spread, the authorship attribution problem has constantly increased its relevance. Nowadays, authorship attribution, for both information retrieval and analysis, has gained great importance in the context of security, trust and copyright preservation.

This work proposes an innovative multi-agent driven machine learning technique that has been developed for authorship attribution. By means of a preprocessing for word-grouping and time-period related analysis of the common lexicon, we determine a bias reference level for the recurrence frequency of the words within analysed texts, and then train a Radial Basis Neural Networks (RBPNN)-based classifier to identify the correct author.

The main advantage of the proposed approach lies in the generality of the semantic analysis, which can be applied to different contexts and lexical domains, without requiring any modification. Moreover, the proposed system is able to incorporate an external input, meant to tune the classifier, and then self-adjust by means of continuous learning reinforcement.

I. INTRODUCTION

Nowadays, the automatic attribution of a text to an author, assisting both information retrieval and analysis, has become an important issue, e.g. in the context of security, trust and copyright preservation. This results from the availability of documents in digital form, and the raising deception possibilities bound to the essence of the digital reproducible contents, as well as the need for new mechanical methods that can organise the constantly increasing amount of digital texts.

During the last decade only, the field of text classification and attribution has undergone new development due to the novel availability of computational intelligence techniques, such as natural language processing, advanced data mining and information retrieval systems, machine learning and artificial intelligence techniques, agent oriented programming, etc. Among such techniques, Computer Intelligence (CI) and Evolutionary Computation (EC) methods have been largely used for optimisation and positioning problems [1], [2]. In [3], agent driven clustering has been used as an advanced solution for some optimal management problems, whereas in [4] such problems are solved for mechatronical module controls. Agent driven artificial intelligence is often used in combination with advanced data analysis techniques in order to create intelligent control systems [5], [6] by means of multi resolution analysis [7]. CI and parallel analysis systems have been proposed in order to support developers, as in [8], [9], [10], [11],

where such a classification and analysis was applied to assist refactoring in large software systems [12], [13], [14], [15].

Moreover, CI and techniques like neural networks (NNs) have been used in [16], [17] in order to model electrical networks and the related controls starting by classification strategies, as well as for other complex physical systems by using several kinds of hybrid NN-based approaches [18], [19], [20], [21]. All the said works use different forms of agent-based modeling and clustering for recognition purposes, and these methods efficiently perform very challenging tasks, where other common computational methods failed or had low efficiency or, simply, resulted as inapplicable due to complicated model underlying the case study. In general, agent-driven machine learning has been proven as a promising field of research for the purpose of text classification, since it allows building classification rules by means of automatic learning, taking as a basis a set of known texts and trying to generalise for unknown ones.

While machine learning and NNs are a very promising field, the effectiveness of such approaches often lies on the correct and precise preprocessing of data, i.e. the definition of semantic categories, affinities and rules used to generate a set of numbers characterising a text sample, to be successively given as input to a classifier. Typical text classification, e.g. by using NNs, takes advantage of topics recognition, however results are seldom appropriate when it comes to classify people belonging to the same social group or who are involved in a similar business (e.g. the classification of: texts from different scientists in the same field of research, the politicians belonging to the same party, texts authored by different people using the same technical jargon).

In our approach we devise a solution for extracting from the analysed texts some characteristics that can express the style of a specific author. Obtaining this kind of information abstraction is crucial in order to create a precise and correct classification system. On the other hand, while data abound in the context of text analysis, a robust classifier should rely on input sets that are compact enough to be apt to the training process. Therefore, some data have to reflect averaged evaluations that concern some anthropological aspects such as the historical period, or the ethnicity, etc. This work satisfies the above conditions of extracting compact data from texts since we use a preprocessing tool for word-grouping and time-period related analysis of the common lexicon. Such a tool computes a bias reference system for the recurrence frequency of the word used in the analysed texts. The main advantage of this choice lies in the generality of the implemented semantical

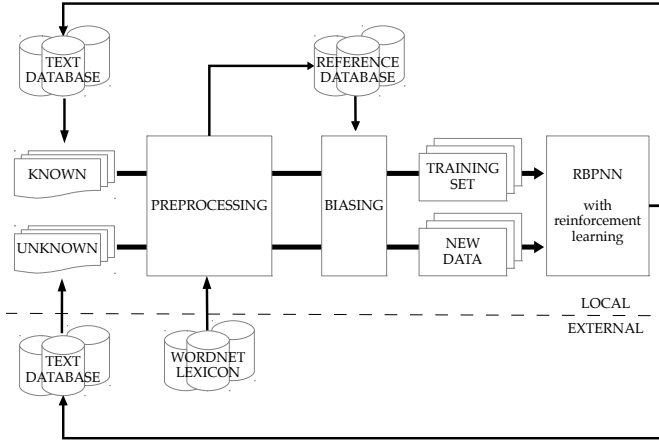


Fig. 1. A general schema of the data flow through the agents of the developed system.

identifier, which can be then applied to different contexts and lexical domains without requiring any modification. Moreover, in order to have continuous updates or complete renewals of the reference data, a statically trained NN would not suffice to the purpose of the work. For these reasons, the developed system is able to self-correct by means of continuous learning reinforcement. The proposed architecture also diminishes the human intervention over time thanks to its self-adaption properties. Our solution comprises three main collaborating *agents*: the first for *preprocessing*, i.e. to extract meaningful data from texts; the second for *classification* by means of a proper Radial Basis NN (RBPNN); and finally, one for *adapting* by means of a feedforward NN.

The rest of this paper is as follows. Section II gives the details of the implemented preprocessing agent based on lexicon analysis. Section III describes the proposed classifier agent based on RBNNs, our introduced modifications and the structure of the reinforcement learning agent. Section IV reports on the performed experiments and the related results. Finally, Section V gives a background of the existing related works, while Section VI draws our conclusions.

II. EXTRACTING SEMANTICS FROM LEXICON

Figure 1 shows the agents for our developed system: a *preprocessing agent* extracts characteristics from given text parts (see text database in the Figure), according to a known set of words organised into groups (see reference database); a *RBPNN agent* takes as input the extracted characteristics, properly organised, and performs the identification on new data, after appropriate training. An additional agent, dubbed *adaptive critic*, shown in Figure 6, dynamically adapts the behaviour of the *RBPNN agent* when new data are available.

Firstly, *preprocessing agent* analyses a text given as input by counting the words that belong to a priori known groups of mutually related words. Such groups contain words that pertain to a given concern, and have been built ad-hoc and according to the semantic relations between words, hence e.g. assisted by the WordNet lexicon¹.

Algorithm 1: Find the group a word belongs to and count occurrences

```

Start,
Import a speech into Text,
Load dictionary into Words,
Load group database into Groups,
thisWord = Text.get();
while thisWord do
    thisGroup = Words.search(thisWord);
    if !thisGroup then
        Load a different Lexicon;
        if Lexicon.exist(thisWord) then
            Words.update();
            Groups.update();
        end
    else
        break;
    end
end
while Text.search(thisWord) do
    Groups.count(thisGroup);
end
thisWord = Text.get();
end
Export Words and Groups,
Stop.
```

The fundamental steps of the said analysis (see also Algorithm 1) are the followings:

- 1) import a single text file containing the speech;
- 2) import word groups from a predefined database, the set containing all words from each group is called dictionary;
- 3) compare each word on the text with words on the dictionary;
- 4) if the word exists on the dictionary then the relevant group is returned;
- 5) if the word has not been found then search the available lexicon;
- 6) if the word exists on the lexicon then the related group is identified;
- 7) if the word is unknown, then a new lexicon is loaded and if the word is found then dictionary and groups are updated;
- 8) search all the occurrences of the word in the text;
- 9) when an occurrence has been found, then remove it from the text and increase the group counter.

Figure 2 shows the UML class diagram for the software system performing the above analysis. Class *Text* holds a text to be analysed; class *Words* represents the known dictionary, i.e. all the known words, which are organised into groups given by class *Groups*; class *Lexicon* holds several dictionaries.

III. THE RBPNN CLASSIFIER AGENT

For the work proposed here, we use a variation on Radial Basis Neural Networks (RBPNN). RBNNs have a topology similar to common FeedForward Neural Networks (FFNN) with BackPropagation Training Algorithms (BPTA): the primary

¹<http://wordnet.princeton.edu>

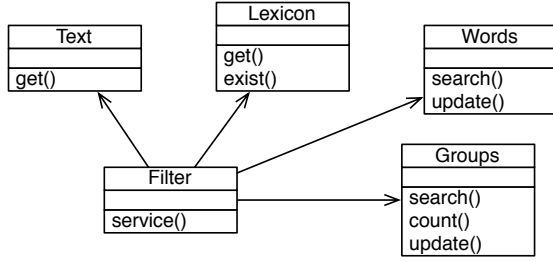


Fig. 2. UML class diagram for handling groups and counting words belonging to a group.

difference only lies in the activation function that, instead of being a sigmoid function or a similar activation function, is a statistical distribution or a statistically significant mathematical function. The selection of transfer functions is indeed decisive for the speed of convergence in approximation and classification problems [22]. The kinds of activation functions used for Probabilistic Neural Networks (PNNs) have to meet some important properties to preserve the generalisation abilities of the ANNs. In addition, these functions have to preserve the decision boundaries of the probabilistic neural networks. The selected RBPNN architecture is shown in Figure 4 and takes advantage from both the PNN topology and the Radial Basis Neural Networks (RBNN) used in [23].

Each neuron performs a weighted sum of its inputs and passes it through a transfer function f to produce an output. This occurs for each neural layer in a FFNN. The network can be perceived as a model connecting inputs and outputs, with the weights and thresholds being free parameters of the model, which are modified by the training algorithm. Such networks can model functions of almost arbitrary complexity with the number of layers and the number of units in each layer determining the function complexity. A FFNN is capable to generalise the model, and to separate the input space in various classes (e.g. in a 2D variable space it is equivalent to the separation of the different semi-planes). In any case, such a FFNN can only create a general model of the entire variable space, while can not insert single set of inputs into categories. On the other hand, a RBNN is capable of clustering the inputs by fitting each class by means of a radial basis function [24], while the model is not general for the entire variable space, it is capable to act on the single variables (e.g. in a 2D variable space it locates closed subspaces, without any inference on the remaining space outside such subspaces).

Another interesting topology is provided by PNNs, which are mainly FFNNs also functioning as Bayesian networks with Fisher Kernels [25]. By replacing the sigmoid activation function often used in neural networks with an exponential function, a PNN can compute nonlinear decision boundaries approaching the Bayes optimal classification [26]. Moreover, a PNN generates accurate predicted target probability scores with a probabilistic meaning (e.g. in the 2D space it is equivalent to attribute a probabilistic score to some chosen points, which in Figure 3 are represented as the size of the points).

Finally, in the presented approach we decided to combine the advantages of both RBNN and PNN using the so called

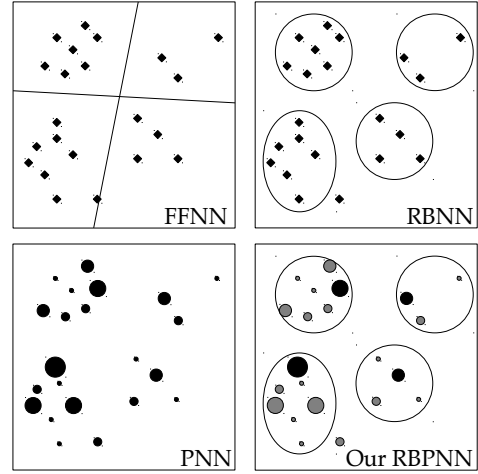


Fig. 3. A comparison between results of several types of NNs, Our RBPNN includes the maximum probability selector module

RBPNN. The RBPNN architecture, while preserving the capabilities of a PNN, due to its topology, then being capable of statistical inference, is also capable of clustering since the standard activation functions of a PNN are substituted by radial basis functions still verifying the Fisher kernel conditions required for a PNN (e.g. such an architecture in the 2D variable space can both locate subspace of points and give to them a probabilistic score). Figure 3 shows a representation of the behaviour for each network topology presented above.

A. The RBPNN structure and topology

In a RBPNN both the input and the first hidden layer exactly match the PNN architecture: the input neurones are used as distribution units that supply the same input values to all the neurones in the first hidden layer that, for historical reasons, are called *pattern units*. In a PNN, each pattern unit performs the dot product of the input pattern vector \mathbf{v} by a weight vector $\mathbf{W}^{(0)}$, and then performs a nonlinear operation on the result. This nonlinear operation gives output $\mathbf{x}^{(1)}$ that is then provided to the following summation layer. While a common sigmoid function is used for a standard FFNN with BPTT, in a PNN the activation function is an exponential, such that, for the j -esime neurone the output is

$$\mathbf{x}_j^{(1)} \propto \exp \left(\frac{\|\mathbf{W}^{(0)} \cdot \mathbf{v}\|}{2\sigma^2} \right) \quad (1)$$

where σ represents the statistical distribution spread.

The given activation function can be modified or substituted while the condition of Parzen (window function) is still satisfied for the estimator \hat{N} . In order to satisfy such a condition some rules must be verified for the chosen window function in order to obtain the expected estimate, which can be expressed as a Parzen window estimate $p(x)$ by means of the kernel K of f in the d -dimensional space S^d

$$p_n(x) = \frac{1}{n} \sum_{i=1}^n \frac{1}{h_n^d} K \left(\frac{x-x_i}{h_n} \right) \quad (2)$$

$$\int_{S^d} K(x) dx = 1$$

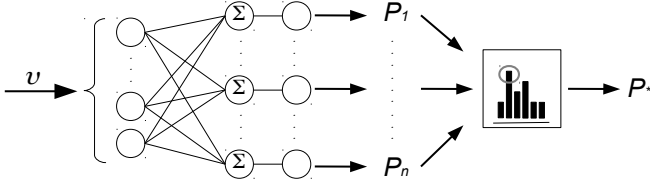


Fig. 4. A representation of a Radial Basis Probabilistic Neural Network with maximum probability selector module

where $h_n \in \mathbb{N}$ is called window width or bandwidth parameter and corresponds to the width of the kernel. In general $h_n \in \mathbb{N}$ depends on the number of available sample data n for the estimator $p_n(x)$. Since the estimator $p_n(x)$ converges in mean square to the expected value $p(x)$ if

$$\begin{aligned} \lim_{n \rightarrow \infty} \langle p_n(x) \rangle &= p(x) \\ \lim_{n \rightarrow \infty} \text{var}(p_n(x)) &= 0 \end{aligned} \quad (3)$$

where $\langle p_n(x) \rangle$ represents the mean estimator values and $\text{var}(p_n(x))$ the variance of the estimated output with respect to the expected values, the Parzen condition states that such convergence holds within the following conditions:

$$\begin{aligned} \sup_x K(x) &< \infty \\ \lim_{|x| \rightarrow \infty} xK(x) &= 0 \\ \lim_{n \rightarrow \infty} h_d^n &= 0 \\ \lim_{n \rightarrow \infty} nh_d^n &= \infty \end{aligned} \quad (4)$$

In this case, while preserving the PNN topology, to obtain the RBPNN capabilities, the activation function is substituted with a radial basis function (RBF); an RBF still verifies all the conditions stated before. It then follows the equivalence between the $\mathbf{W}^{(0)}$ vector of weights and the centroids vector of a radial basis neural network, which, in this case, are computed as the statistical centroids of all the input sets given to the network.

We name f the chosen radial basis function, then the new output of the first hidden layer for the j -esime neurone is

$$\mathbf{x}_j^{(1)} \triangleq f\left(\frac{\|\mathbf{v} - \mathbf{W}^{(0)}\|}{\beta}\right) \quad (5)$$

where β is a parameter that is intended to control the distribution shape, quite similar to the σ used in (1).

The second hidden layer in a RBPNN is identical to a PNN, it just computes weighted sums of the received values from the preceding neurones. This second hidden layer is called indeed summation layer: the output of the k -esime summation unit is

$$\mathbf{x}_k^{(2)} = \sum_j \mathbf{W}_{jk} \mathbf{x}_j^{(1)} \quad (6)$$

where \mathbf{W}_{jk} represents the weight matrix. Such weight matrix consists of a weight value for each connection from the j -esime pattern units to the k -esime summation unit. These summation

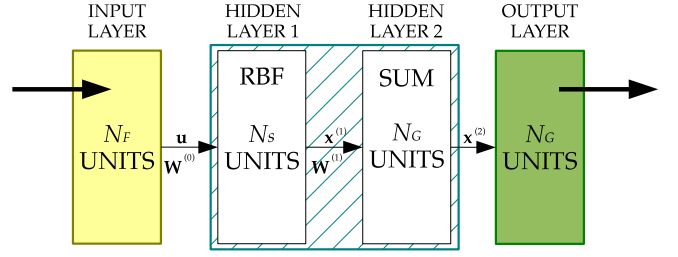


Fig. 5. Setup values for the proposed RBPNN: N_F is the number of considered lexical groups, N_S the number of analysed texts, and N_G is the number of people that can possibly be recognised as authors.

units work as in the neurones of a linear perceptron network. The training for the output layer is performed as in a RBNN, however since the number of summation units is very small and in general remarkably less than in a RBNN, the training is simplified and the speed greatly increased [27].

The output of the RBPNN (as shown in Figure 4) is given to the maximum probability selector module, which effectively acts as a one-neuron output layer. This selector receives as input the probability score generated by the RBPNN and attributes to one author only the analysed text, by selecting the most probable author, i.e. the one having the maximum input probability score. Note that the links to this selector are weighted (with weights adjusted during the training), hence the actual input is the product between the weight and the output of the summation layer of the RBPNN.

B. Layer size for a RBPNN

The devised topology enables us to distribute to different layers of the network different parts of the classification task. While the pattern layer is just a nonlinear processing layer, the summation layer selectively sums the output of the first hidden layer. The output layer fulfills the nonlinear mapping such as classification, approximation and prediction. In fact, the first hidden layer of the RBPNN has the responsibility to perform the fundamental task expected from a neural network [28]. In order to have a proper classification of the input dataset, i.e. of analysed texts to be attributed to authors, the size of the input layer should match the exact number N_F of different lexical groups given to the RBPNN, whereas the size of the pattern units should match the number of samples, i.e. analysed texts, N_S . The number of the summation units in the second hidden layer is equal to the number of output units, these should match the number of people N_G we are interested in for the correct recognition of the speakers (Figure 5).

C. Reinforcement learning

In order to continuously update the reference database for our system, a statically trained NN would not suffice for the purpose of the work. Since the aim of the presented system is having an expanding database of text samples for classification and recognition purpose, the agent driven identification should dynamically follow the changes in such a database. When a new entry is made then the related feature set and biases change, it implies that also the RBPNN should be properly managed in order to ensure a continuous adaptive control for reinforcement learning. Moreover, for the considered domain

it is desirable that a human supervisor supply suggestions, especially when the system starts working. The human activities are related to the supply of new entries into the text sample database, and to the removal of misclassifications made by the RBPNN.

We used a supervised control configuration (see Figure 6), where the external control is provided by the actions and choices of a human operator. While the RBPNN is trained with a classical backpropagation learning algorithm, it is also embedded into an actor-critic reinforcement learning architecture, which back propagates learning by evaluating the correctness of the RBPNN-made choices with respect to the real word.

Let ξ be the error function, i.e. $\xi = 0$ for the results supported by human verification, or the vectorial deviance for the results not supported by a positive human response. This assessment is made by an agent named *Critic*. We consider the filtering step for the RBPNN output, to be both: *Critic*, i.e. a human supervisor acknowledging or rejecting RBPNN classifications; or *Adaptive critic*, i.e. an agent embedding a NN that in the long run simulates the control activity made by the human *Critic*, hence decreasing human control over time. *Adaptive critic* needs to learn, and this learning is obtained by a modified backpropagation algorithm using just ξ as error function. Hence, *Adaptive critic* has been implemented by a simple feedforward NN trained, by means of a traditional gradient descent algorithm so that the weight modification Δw_{ij} is

$$\Delta w_{ij} = -\mu \frac{\partial \xi}{\partial w_{ij}} = -\mu \frac{\partial \xi}{\partial \tilde{f}_i} \frac{\partial \tilde{f}_i}{\partial \tilde{u}_i} \frac{\partial \tilde{u}_i}{\partial w_{ij}} \quad (7)$$

The \tilde{f}_i is the activation of i -esime neuron, \tilde{u}_i is the i -esime input to the neurone weighted as

$$\tilde{u}_i = \sum_j w_{ij} \tilde{f}_j(\xi_i) \quad (8)$$

The result of the adaptive control determines whether to continue the training of the RBPNN with new data, and whether the last training results should be saved or discarded. At runtime this process results in a continuous adaptive learning, hence avoiding the classical problem of NN polarisation and overfitting. Figure 6 shows the developed learning system reinforcement. According to the literature [29], [30], [31], [32], straight lines represent the data flow, i.e. training data fed to the RBPNN, then new data inserted by a supervisor, and the output of the RBPNN sent to the Critic modules also by means of a delay operator z^{-1} . Functional modifications operated within the system are represented as slanting arrows, i.e. the choices made by a human supervisor (*Critic*) modify the *Adaptive critic*, which adjust the weight of its NN; the combined output of *Critic* and *Adaptive critic* determines whether the RBPNN should undergo more training epochs and so modify its weights.

IV. EXPERIMENTAL SETUP

The proposed RBPNN architecture has been tested using several text samples collected from public speeches of different people both from the present and the past era. Each text sample has been given to the *preprocessing agent* that extract some

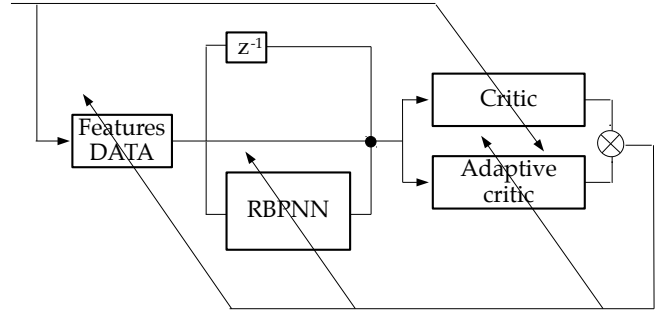


Fig. 6. The adopted supervised learning model reinforcement. Slanting arrows represent internal commands supplied in order to control or change the status of the modules, straight arrows represent the data flow along the model, z^{-1} represents a time delay module which provides 1-step delayed outputs.

characteristics (see Section II), then such results have been given to the *classification agent*. The total number of text samples was 344, and we used 258 of them for training the *classification agent* and 86 for validation. The text samples, both for training and validation, were from different persons that have given a speech (from A. Einstein to G. Lewis), as shown in Figure 7.

Given the flexible structure of the implemented learning model, the word groups are not fixed and can be modified, added or removed over time by an external tuning activity. By using the count of words in a group, instead of a word-by-word counts, the multi-agent system realises a statistically driven classifier that identifies the main semantic concerns regarding the text samples, and then, attributes such concerns to the most probable person.

The relevant information useful in order to recognise the author of the speech is usually largely spread over a certain number of word groups that could be indication of the cultural extraction, heritage, field of study, professional category etc. This implies that we can not exclude any word group, a priori, while the RBPNN could learn to automatically enhance the relevant information in order to classify the speeches.

Figure 7-left shows an example of the classifier performances for results generated by the RBPNN (before the filter implemented by the probabilistic selector). Since the RBPNN results have a probability between 0 and 1, then the shown performance is 0 when a text was correctly attributed (or not attributed) to a specific person. Figure 7-right shows the performances of the system when including the probabilistic selector. For this case, a boolean selection is involved, then correct identifications are represented as 0, false positive identifications as -1 (black marks), and missed identifications as $+1$ (white marks).

For validation purposes, Figure 7-(left and right) shows results according to e :

$$e = y - \tilde{y} \quad (9)$$

where e identifies the performance, \tilde{y} the classification result, and y the expected result. Lower e (negative values) identify an excess of confidence in the attribution of a text to a person, while greater e (positive values) identify a lack of confidence in that sense.

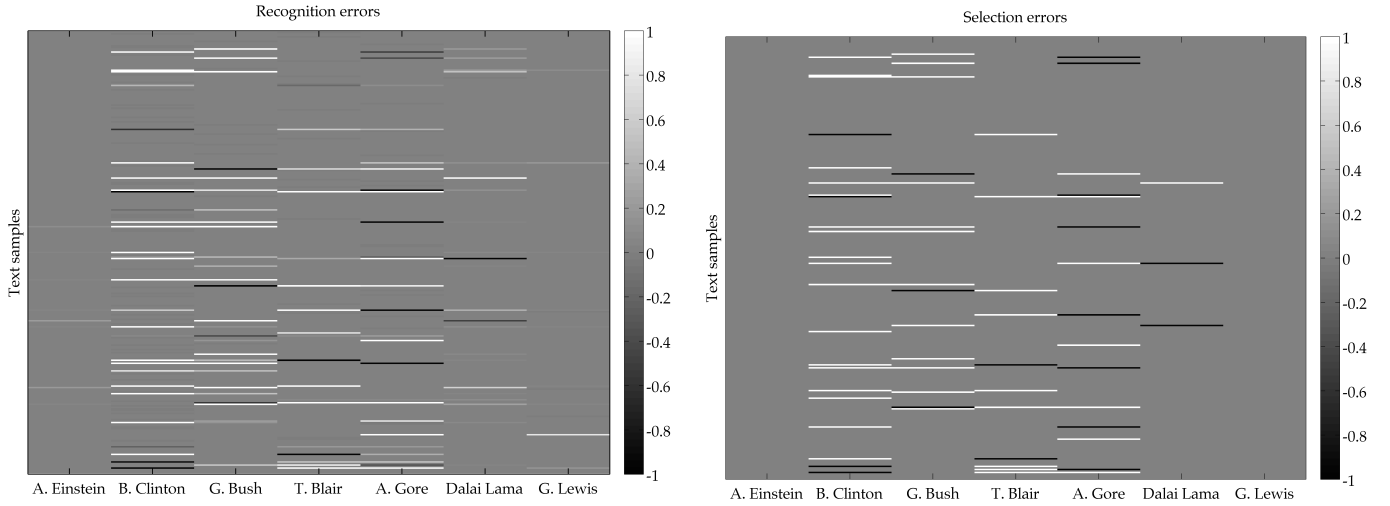


Fig. 7. The obtained performance for our classification system before (left) and after (right) the maximum probability selector choice. The mean grey color represents the correct classifications, while white color represents missed classification and black color false classifications.

The system was able to correctly attribute the text to the proper author with only a 20% of missing assignments.

V. RELATED WORKS

Several generative models can be used to characterise datasets that determine properties and allow grouping data into classes. Generative models are based on stochastic block structures [33], or on ‘Infinite Hidden Relational Models’ [34], and ‘Mixed Membership Stochastic Blockmodel’ [35]. The main issue of class-based models is the type of relational structure that such solutions are capable to describe. Since the definition of a class is attribute-dependent, generally the reported models risk to replicate the existing classes for each new attribute added. E.g. such models would be unable to efficiently organise similarities between the classes ‘cats’ and ‘dogs’ as child classes of the more general class ‘mammals’. Such attribute-dependent classes would have to be replicated as the classification generates two different classes of ‘mammals’: the class ‘mammals as cats’ and the class ‘mammals as dogs’. Consequently, in order to distinguish between the different races of cats and dogs, it would be necessary to further multiply the ‘mammals’ class for each one of the identified race. Therefore, such models quickly lead to an explosion of classes. In addition, we would either have to add another class to handle each specific use or a mixed membership model, as for crossbred species.

Another paradigm concerns the ‘Non-Parametric Latent Feature Relational Model’ [36], i.e. a Bayesian nonparametric model in which each entity has boolean valued latent features that influence the model’s relations. Such relations depend on well-known covariant sets, which are neither explicit or known in our case study at the moment of the initial analysis.

In [37], the authors propose a sequential forward feature selection method to find the subset of features that are relevant to a classification task. This approach uses novel estimation of the conditional mutual information between candidate feature and classes, given a subset of already selected features used as a classifier independent criterion for evaluating feature subsets.

In [38], data from the charge-discharge simulation of lithium-ions battery energy storage are used for classification purposes with recurrent NNs and PNNs by means of a theoretical framework based on signal theory.

While showing the effectiveness of the neural network based approaches, in our case study classification results are given by means of a probability, hence the use of a RBPNN, and an on-line training achieved by reinforcement learning.

VI. CONCLUSION

This work has presented a multi-agent system, in which an agent analyses fragments of texts and another agent consisting of a RBPNN classifier, performs probabilistic clustering. The system has successfully managed to identify the most probable author among a given list for the examined text samples. The provided identification can be used in order to complement and integrate a comprehensive verification system, or other kinds of software systems trying to automatically identify the author of a written text. The RBPNN classifier agent is continuously trained by means of reinforcement learning techniques in order to follow a potential correction provided by an human supervisor, or an agent that learns about supervision. The developed system was also able to cope with new data that are continuously fed into the database, for the adaptation abilities of its collaborating agents and their reasoning based on NNs.

ACKNOWLEDGMENT

This work has been supported by project PRIME funded within POR FESR Sicilia 2007-2013 framework and project PRISMA PON04a2 A/F funded by the Italian Ministry of University and Research within PON 2007-2013 framework.

REFERENCES

- [1] C. Napoli, G. Pappalardo, E. Tramontana, Z. Marszałek, D. Połap, and M. Woźniak, “Simplified firefly algorithm for 2d image key-points search,” in *IEEE Symposium Series on Computational Intelligence*. IEEE, 2014.

- [2] M. Gabryel, M. Woźniak, and R. K. Nowicki, "Creating learning sets for control systems using an evolutionary method," in *Proceedings of Artificial Intelligence and Soft Computing (ICAISC)*, ser. LNCS, vol. 7269. Springer, 2012, pp. 206–213.
- [3] F. Bonanno, G. Capizzi, A. Gagliano, and C. Napoli, "Optimal management of various renewable energy sources by a new forecasting method," in *Proceedings of International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM)*. IEEE, 2012, pp. 934–940.
- [4] A. Nowak and M. Woźniak, "Analysis of the active module mechatronical systems," in *Proceedings of Mechanika - ICM*. Kaunas, Lietuva: Kaunas University of Technology Press, 2008, pp. 371–376.
- [5] C. Napoli, G. Pappalardo, and E. Tramontana, "A hybrid neuro-wavelet predictor for qos control and stability," in *Proceedings of AI*IA: Advances in Artificial Intelligence*. Springer, 2013, pp. 527–538.
- [6] F. Bonanno, G. Capizzi, G. L. Sciuto, C. Napoli, G. Pappalardo, and E. Tramontana, "A novel cloud-distributed toolbox for optimal energy dispatch management from renewables in igss by using wrnn predictors and gpu parallel solutions," in *Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), 2014 International Symposium on*. IEEE, 2014, pp. 1077–1084.
- [7] A. Nowak and M. Woźniak, "Multiresolution derives analysis of module mechatronical systems," *Mechanika*, vol. 6, no. 74, pp. 45–51, 2008.
- [8] C. Napoli, G. Pappalardo, and E. Tramontana, "Using modularity metrics to assist move method refactoring of large systems," in *Proceedings of International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)*. IEEE, 2013, pp. 529–534.
- [9] G. Pappalardo and E. Tramontana, "Suggesting extract class refactoring opportunities by measuring strength of method interactions," in *Proceedings of Asia Pacific Software Engineering Conference (APSEC)*. IEEE, December 2013.
- [10] E. Tramontana, "Automatically characterising components with concerns and reducing tangling," in *Proceedings of Computer Software and Applications Conference (COMPSAC) Workshop QUORS*. IEEE, July 2013. DOI: 10.1109/COMPSACW.2013.114, pp. 499–504.
- [11] C. Napoli, G. Pappalardo, and E. Tramontana, "Improving files availability for bittorrent using a diffusion model," in *IEEE 23rd International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises - WETICE 2014*, June 2014, pp. 191–196.
- [12] R. Giunta, G. Pappalardo, and E. Tramontana, "Aspects and annotations for controlling the roles application classes play for design patterns," in *Proceedings of Asia Pacific Software Engineering Conference (APSEC)*. IEEE, December 2011, pp. 306–314.
- [13] A. Calvagna and E. Tramontana, "Delivering dependable reusable components by expressing and enforcing design decisions," in *Proceedings of Computer Software and Applications Conference (COMPSAC) Workshop QUORS*. IEEE, July 2013. DOI: 10.1109/COMPSACW.2013.113, pp. 493–498.
- [14] R. Giunta, G. Pappalardo, and E. Tramontana, "AODP: refactoring code to provide advanced aspect-oriented modularization of design patterns," in *Proceedings of Symposium on Applied Computing (SAC)*. ACM, 2012.
- [15] E. Tramontana, "Detecting extra relationships for design patterns roles," in *Proceedings of AsianPloP*, March 2014.
- [16] G. Capizzi, C. Napoli, and L. Paternò, "An innovative hybrid neuro-wavelet method for reconstruction of missing data in astronomical photometric surveys," in *Proceedings of Artificial Intelligence and Soft Computing (ICAISC)*. Springer, 2012, pp. 21–29.
- [17] F. Bonanno, G. Capizzi, G. L. Sciuto, C. Napoli, G. Pappalardo, and E. Tramontana, "A cascade neural network architecture investigating surface plasmon polaritons propagation for thin metals in openmp," in *Proceedings of Artificial Intelligence and Soft Computing (ICAISC)*, ser. LNCS, vol. 8467. Springer, 2014, pp. 22–33.
- [18] C. Napoli, F. Bonanno, and G. Capizzi, "Exploiting solar wind time series correlation with magnetospheric response by using an hybrid neuro-wavelet approach," *Proceedings of the International Astronomical Union*, vol. 6, no. S274, pp. 156–158, 2010.
- [19] G. Capizzi, F. Bonanno, and C. Napoli, "Hybrid neural networks architectures for soc and voltage prediction of new generation batteries storage," in *Proceedings of International Conference on Clean Electrical Power (ICCEP)*. IEEE, 2011, pp. 341–344.
- [20] C. Napoli, F. Bonanno, and G. Capizzi, "An hybrid neuro-wavelet approach for long-term prediction of solar wind," *IAU Symposium*, no. 274, pp. 247–249, 2010.
- [21] G. Capizzi, F. Bonanno, and C. Napoli, "A new approach for lead-acid batteries modeling by local cosine," in *Power Electronics Electrical Drives Automation and Motion (SPEEDAM), 2010 International Symposium on*, June 2010, pp. 1074–1079.
- [22] W. Duch, "Towards comprehensive foundations of computational intelligence," in *Challenges for Computational Intelligence*. Springer, 2007, pp. 261–316.
- [23] G. Capizzi, F. Bonanno, and C. Napoli, "Recurrent neural network-based control strategy for battery energy storage in generation systems with intermittent renewable energy sources," in *Proceedings of International Conference on Clean Electrical Power (ICCEP)*. IEEE, 2011, pp. 336–340.
- [24] S. Haykin, *Neural Networks - A comprehensive foundation*. Prentice Hall, 2004.
- [25] S. Mika, G. Ratsch, W. Jason, B. Scholkopf, and K.-R. Muller, "Fisher discriminant analysis with kernels," in *Proceedings of the Signal Processing Society Workshop. Neural networks for signal processing IX*. IEEE, 1999.
- [26] D. F. Specht, "Probabilistic neural networks," *Neural networks*, vol. 3, no. 1, pp. 109–108, 1990.
- [27] H. Deshuang and M. Songde, "A new radial basis probabilistic neural network model," in *Proceedings of Conference on Signal Processing*, vol. 2. IEEE, 1996.
- [28] W. Zhao, D.-S. Huang, and L. Guo, "Optimizing radial basis probabilistic neural networks using recursive orthogonal least squares algorithms combined with micro-genetic algorithms," in *Proceedings of Neural Networks*, vol. 3. IEEE, 2003.
- [29] D. V. Prokhorov, R. A. Santiago, and D. C. W. II, "Adaptive critic designs: A case study for neurocontrol," *Neural Networks*, vol. 8, no. 9, pp. 1367 – 1372, 1995. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/0893608095000429>
- [30] H. Javaherian, D. Liu, and O. Kovalenko, "Automotive engine torque and air-fuel ratio control using dual heuristic dynamic programming," in *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, 2006, pp. 518–525.
- [31] B. Widrow and M. Lehr, "30 years of adaptive neural networks: perceptron, madaline, and backpropagation," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1415–1442, Sep 1990.
- [32] J.-W. Park, R. Harley, and G. Venayagamoorthy, "Adaptive-critic-based optimal neurocontrol for synchronous generators in a power system using mlp/rbf neural networks," *IEEE Transactions on Industry Applications*, vol. 39, no. 5, pp. 1529–1540, Sept 2003.
- [33] K. Nowicki and T. A. B. Snijders, "Estimation and prediction for stochastic blockstructures," *Journal of the American Statistical Association*, vol. 96, no. 455, pp. 1077–1087, 2001.
- [34] Z. Xu, V. Tresp, K. Yu, and H. peter Krieger, "Infinite hidden relational models," in *In Proceedings of International Conference on Uncertainty in Artificial Intelligence (UAI)*, 2006.
- [35] E. M. Airoldi, D. M. Blei, E. P. Xing, and S. E. Fienberg, "Mixed membership stochastic block models," in *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, 2009.
- [36] K. Miller, T. Griffiths, and M. Jordan, "Nonparametric latent feature models for link prediction," in *Advances in neural information processing systems (NIPS)*. Curran Associates, Inc., 2009, vol. 22, pp. 1276–1284.
- [37] J. Novovičová, P. Somol, M. Haindl, and P. Pudil, "Conditional mutual information based feature selection for classification task," in *Progress in Pattern Recognition, Image Analysis and Applications*. Springer, 2007, pp. 417–426.
- [38] F. Bonanno, G. Capizzi, and C. Napoli, "Some remarks on the application of rnn and prnn for the charge-discharge simulation of advanced lithium-ions battery energy storage," in *Proceedings of International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM)*. IEEE, 2012, pp. 941–945.